# 広域処理分散自動化における スケーラブルな外部環境変化への追従手法の提案

福田茂紀 †1　福井誠之 †1　佐野健 †1　久保田真 †1　野村佳秀 †1
佐々木和雄 †1　阿比留健一 †1

ネットワークに接続する機器の増加に伴って，データセンター集約型システムの通信トラフィック増大やレスポンス低下の問題がより深刻になっているが，処理の一部をネットワーク上に分散させる試みは，アプリケーション開発者や運用者に対する設計と運用の負担が大きく，外部環境に対して適切な分散状態を維持するのが困難であった．本論文では，運用者が設定する簡単な分散ポリシーを元に，自動的に処理やデータを最適な場所に配備し，環境変化に応じてスケーラブルに再配備する分散サービス基盤を提案し，試作システムによって外部環境追従性能を評価して報告する．

# Proposal of the scalable method of following to environmental changes for automatic wide area process distribution

SHIGEKI FUKUTA†1　MASAYUKI FUKUI†1　TAKESHI SANO†1
MAKOTO KUBOTA†1　YOSHIHIDE NOMURA†1
KAZUO SASAKI†1　KENICHI ABIRU†1

## 1. はじめに

近年，端末機器やセンサーの高性能化と低価格化が進み，あらゆる機器がネットワークに接続される Internet of Things (IoT) の実現が近づいている．例えば各家庭の使用電力や全国の自動販売機の購買情報のような，実世界の情報を機器から自動的に収集して活用する M2M サービスが，IoT の一部分として実現され始めている．これらの収集データ発生源の増加が続くと，従来のデータセンターにアプリケーションを集中する構成では，センターに集まるデータや機器へ配信するデータの量が増大し，通信コスト増大と輻輳によるレスポンス低下が問題となる．このため，機器の傍でフィルタリングや集計などの事前処理を行うことで，通信トラフィックを削減し，処理レスポンスを高速化する，広域分散システムが期待されている．しかし，適切な事前処理を適切な場所で実行し続けるためには，多数の処理実行環境と分散処理の状態を把握して判断せねばならず，大きな労力が必要となる．

そこで本論文では，運用者が設定する簡単な分散ポリシーから，自動的に処理やデータを最適な場所に配備し，環境変化に応じてスケーラブルに再配備する分散サービス基盤を提案する．以下，2 章では広域分散システムの背景について述べ，3 章で既存の広域分散システムの運用面の課題を説明する．そして，4 章で提案システムの設計を，5 章でその特徴を説明し，6 章で提案システムの評価結果について述べて，7 章で本研究の成果をまとめる．

†1 (株)富士通研究所

## 2. 広域分散システムの背景

前節で述べたような，機器数の増大に伴う通信トラフィックの問題に対してアプリケーションの応答性能を保つためには，データセンターや通信網の設備増強が必要となり，システム規模拡大の妨げとなっている．現実的なコストで規模を拡大可能にするには，処理の内容や収集データの特質に合わせた，効率的なネットワーク利用が不可欠である．

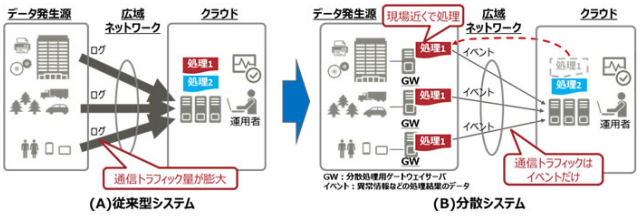



図 1 広域分散システムの考え方

そこで図 1 に示すように，センターに集中する処理やデータの一部を，機器・デバイスの近くに存在する処理ノードに分散して処理させることで，通信トラフィック削減やレスポンス向上を実現させる，より広範囲な処理・データの分散システムが注目されている．

図 2 に機器監視情報の収集分析の一事例を示す．電力メーターのような監視対象機器は，データ発生源として各地の拠点 (フロント領域) に置かれている．フロント領域や広域ネットワーク上の処理ノードで，電力値のうち前回から変化が無いデータをフィルタする処理や，一定時間毎の集計処理などの部分処理を実行することで，センターへの

トラフィックを減らし，迅速な処理が必要な結果を優先して送ることで，通信コスト削減とレスポンス向上を実現できる．

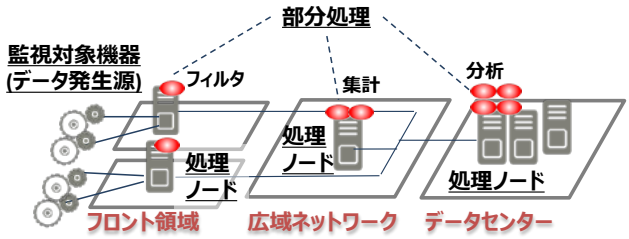



図 2 機器監視情報収集分析の広域分散

## 3. 広域分散システム運用の課題

このような広域分散システムが置かれる環境には，日々新しいデバイスが登場し，増加するユーザや機器をネットワークに接続するため，機器とネットワーク環境は常に変化している．また，これらの機器からの情報を利用するアプリケーションと収集データの内容も，収集・蓄積したデータの分析から得られた知見をフィードバックして，頻繁に更新されていく．これまでは，ネットワークの効率的な利用のため，アプリケーションをどこに分散させるかを，アプリロジックやネットワークトポロジーを元に人間が決定していた．このため，複雑な分散構成や急激なデータ量変化への迅速な対応が困難であり，また設計期間が長期化して運用コストが増加するという課題があった．さらに機器や処理ノード数が増加し続ければ，人間の能力では管理しきれなくなる．これらの課題を解決するため，機器からセンターまでのシステム全体の状況変化を検知し，周辺環境，ネットワーク，アプリケーションの変化に応じて，自動的に分散状態を追従させていくシステムが重要となる．

過去の処理の分散実行ノード最適化の関連研究として，Shneidman らが，Stream Based Overlay Networks: SBON の研究において，ストリーム処理の実行ノードの最適化の研究を行い，処理ノード間の通信レイテンシと CPU 負荷を 3 次元コスト空間にマッピングし，処理の実行ノードを決定する手法を提案している[1]．また，Borealis SPE [2][3]はブランダイス大学他が開発した分散ストリーム処理システムであり，Hwang らが北米の広い範囲に分散した処理ノード上でストリーム処理を実行させている[4]．筆者らも，アプリケーションをデータの集約特性に合わせて分割し，集約のキーとなる値の流れる経路に合わせた分散配備手法を提案している[5]．

しかし広域分散システムでは，多様なネットワーク環境下にある大量のノード管理を行い，収集経路や収集データ量の偏りと変動に合わせて分散状態を最適に維持し続けることが重要である．先に示した既存研究では，通信コストと処理スループットを最適化するための，分散配備場所を計算する機能は設けられていたが，システム全体の監視と変化に追従する自動分散の仕組みが不十分であった．

## 4. 分散サービス基盤

この章では，提案する分散サービス基盤のアーキテクチャ，構成の概要と，アーキテクチャで定義するアプリケーション，分散サービス基盤，インフラの 3 層の役割について説明する．

### 4.1 分散サービス基盤のアーキテクチャ

本論文で提案する分散サービス基盤は，開発者が定義したアプリケーションを，運用者が定義する配備ポリシーに従って，収集ネットワーク上の処理ノードへ自動分散させる．そして，機器やデータ量の増減などの変化を検知し，最適な配備先ノードの計算と再配備を繰り返すことで，実行環境の変化に追従して最適な分散配備を維持する．

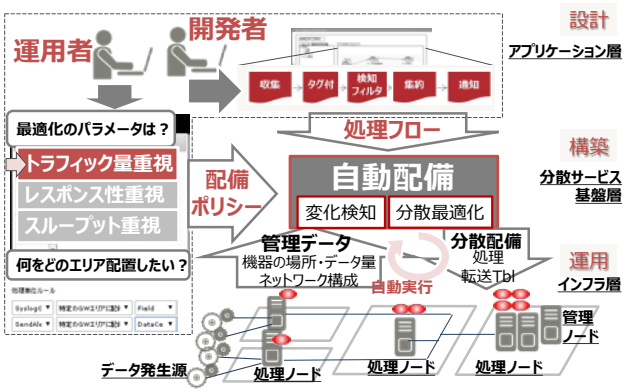



図 3 変化に追従した分散配備

分散サービス基盤のアーキテクチャは，開発者が作成するアプリケーション層 (図 3 上段) と，ネットワーク上の処理ノードでアプリケーションを分散実行するインフラ層 (図 3 下段) と，運用者の指示に従ってアプリケーションの分割と実行場所の決定と分散指示を行う基盤層 (図 3 中段) の 3 層で構成する．中間の分散サービス基盤層が，上下のアプリ状態とインフラ環境を監視して，最適なノードに処理/データを移動させ続けることで，環境変化に追従した分散サービス環境を最適に維持する．

アプリケーション層は，サービスを構成するアプリケーションを，処理部品のフローの形で分散サービス基盤に与え，各部品の実行特性を定義する．インフラ層は，処理ノードとネットワークの状況と，ノード上で実行中の処理状態を表す監視データを，分散サービス基盤層に与える．分散サービス基盤層は，上記のように与えられたアプリ定義・実行状態とインフラ環境の情報を元に，サービス運用者が与える配備ポリシーを満たす最適な配備組み合わせを

計算する．そして，その組み合わせに応じてアプリケーションとデータをインフラ層の処理ノードに配信し，収集データや処理結果を次の部分処理につなげる転送ルールを設定することで，分散処理実行環境を構築する．

### 4.2 基盤構成概要

自律分散型の管理システムは，単純なスケーラビリティの面では有効であるが，システム全体の状態を管理者が把握するのが難しい．分散サービス基盤では，基盤と開発者がサービス全体を俯瞰的に把握しやすくするため，センターによる一括管理型の構成とし，分散システム全体の管理機能を持つ管理ノードをデータセンター側に置く．

管理ノードと収集ノードの機能構成を示したのが図 4 である．管理ノードは，分散システムの管理に必要な情報を一元管理する．管理対象として，入力情報 (処理フローと配備ポリシー) と管理データ (ノード及びネットワーク情報と，分散処理状態の監視データ) を蓄積している．

処理ノードは，分散処理実行とデータ転送に加えて，処理状態の監視部品を設け，処理状態に追従した分散配備を実現する．監視対象は，処理部品間の通信量やレスポンス時間をはじめとした処理の実行状態である．また，各ノードからは検出しにくいネットワーク情報や VM システム情報は，下位のインフラ層の管理システムから取得する．

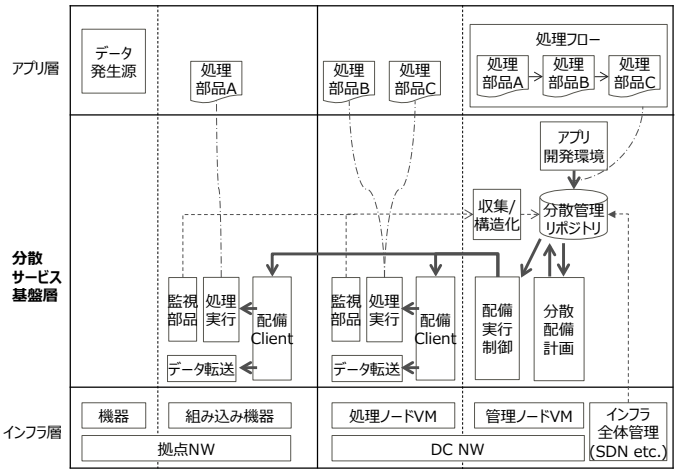



図 4 分散サービス基盤機能構成

以下の節では，これまでに説明したアプリケーション，分散サービス基盤，インフラの 3 層の役割と動きについて，個別に説明する．

### 4.3 アプリケーション層

分散サービス基盤は，配備対象アプリケーションを部分処理に分け，さらに各部分処理が入力とするデータの集約単位に分けて実行先ノードを決定し，分散配備を実行する．

前者については，アプリケーションの部分処理をイベントドリブンな処理部品で構成し，開発者がこの処理部品をつなげた処理フローの形でアプリケーションを定義する．基盤は，各部品単位で処理ノードへの配備と管理を行う．

後者については，各部品が入力データを ID やタグなどの特定の属性によって集約して処理をする場合に，属性値毎の処理に，独立に処理ノードを割り当てる．分散サービス基盤はこの集約属性の値を実行時に識別して，分割とノードの割り当てを行う．属性値によらず全入力を集約する処理部品は，単一の処理として一つの処理ノードを割り当てる．この部品ごとの集約単位の有無や対象属性は，基盤による解析，または管理者の指定によって，処理部品のパラメータとして定義する．集約属性の実例としては，平均値計算処理に於いて「ユーザ ID 毎」に平均値を求める，最大値取得処理に於いて「地域名毎」に最大値を取得して通知する，のような定義を行う．

開発者は，図 5 に示すような開発環境で，部品をドラッグ＆ドロップして，パラメータを定義し，部品を結合することで処理フローを作成する．この開発環境で構築された処理フローを，管理ノードの管理リポジトリに登録することで，分散サービス基盤から利用可能にする．

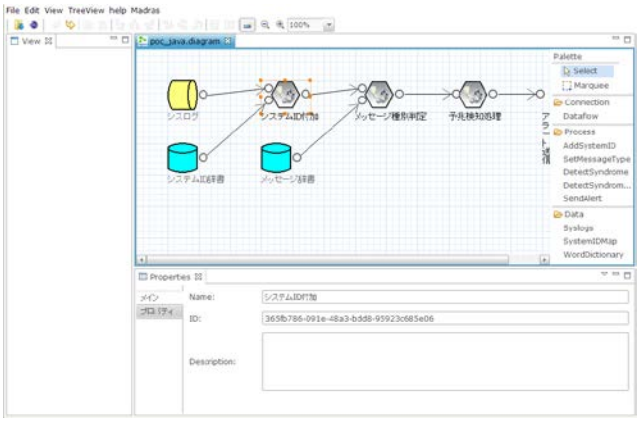


図 5 処理フロー作成画面

### 4.4 分散サービス基盤層

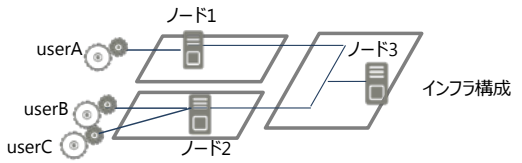
分散サービス基盤は，運用者が定義する，アプリケーションをどのように分散すべきかの指針 (配備ポリシー) に従って，分散配備先を計算する．設定可能なポリシーを，以下に示す．

- 分散配備の優先項目: トラフィック量最小を目指すか，処理結果のレスポンス時間高速化を目指すか，を選択
- 処理やノードごとの配備条件: 処理毎の特性によって，配備可能な処理ノードの条件を記述
  例) センサーへのローカルアクセスを必要とする処理はセンサーと同一セグメントに配備する

分散サービス基盤の分散配備最適化処理は，優先項目に対応する評価関数と，配備先条件に対応する配備制約を作成する．そして前節で分割した部分処理に対する処理ノー

ドの割り当て組み合わせから，配備制約を満たす組み合わせを生成し，評価関数の値 (評価値) が最善のものを探索する．図 6 に，生成するノード割り当ての組み合わせの例を示す．この図の例では，ユーザ ID 毎に行う収集処理と，部署毎に行う検知処理と検知結果の通知処理の 3 種 6 つの部分処理が存在する．この部分処理を，図の上部に示すインフラ構成にある 3 つの処理ノードに割り振る．

分散サービス基盤が利用する評価関数や制約条件と配備ポリシー設定の組み合わせは，分散サービスの特質に合わせて追加・更新することができる．




| 処理 / 組み合わせ | 収集 (ユーザID単位) | | | 検知 (部署単位) | | 通知 |
|---|---|---|---|---|---|---|
| | userA | userB | userC | 部署X | 部署Y | |
| 組み合わせ1 | ノード1 | ノード2 | ノード2 | ノード1 | ノード3 | ノード3 |
| 組み合わせ2 | ノード1 | ノード2 | ノード2 | ノード3 | ノード2 | ノード3 |
| 組み合わせ3 | ノード1 | ノード2 | ノード2 | ノード3 | ノード3 | ノード3 |
| ..... | | | | | | |

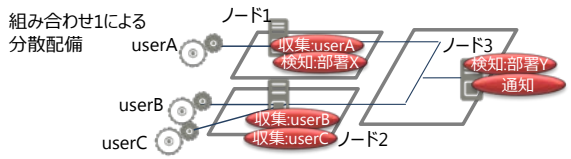



図 6 部分処理のノード割り当て組み合わせ

### 4.5 インフラ層

分散サービス基盤層は，処理の配備先ノードを最適化する際に，ネットワークと処理ノードの情報をインフラ層の管理機能から取得して利用する．

最適化に利用するネットワーク性能情報，即ちノード間のスループット，コスト，レイテンシの情報は，インフラ層のネットワーク管理機構で管理・制御されており，分散サービス基盤はその機構の管理 API を通してこの情報を取得する．一方，処理ノードの処理能力，メモリやストレージなどの情報も，VM マネージャに代表されるインフラ層の管理機構から取得するが，物理的なサーバや組み込み機器を利用して性能管理を行う主体が無い場合は，各ノードに配置した監視部品を通して取得する．

## 5. 分散サービスの大規模化に向けた手法

この章では，4.4 節で説明した分散サービス基盤の配備最適化処理の高速化と，管理データ収集の効率化によって，大規模なシステムに於いても環境の変化に対して柔軟で，素早い追従を実現させる技術について述べる．

### 5.1 配備先計算の高速化

4.4 節で説明したように，分散配備最適化処理は部分処理に実行ノードを割り当てる組み合わせの中から，評価関数を最適にする組み合わせを探索する．このような組み合わせ最適化問題は，整数計画問題に属し，一般に NP 困難な性質を持つ．このため，処理部品数や規模に伴って計算時間が膨大になってしまう．

そこで，分散対象のアプリケーションがデータをセンターに収集する途上で処理を行うという特徴を使い，以下の 2 つのヒューリスティックスを組み込み，処理の特性に応じてこれらの戦略を切り替えて配備先を探索することで，高速に準最適解を探索可能にした．

① 最短経路上のノードへの絞り込み探索

一般論として，データ発生源から最終収集場所までの最短経路上に処理を配備できれば，データがネットワーク中を遠回りすることなく，総トラフィックを削減できる．また，集約対象の入力データが最短経路上で多く合流するノードに処理を配備できれば，集約のために迂回するトラフィック量を削減することができる．

そのため，各処理の探索開始場所をデータ発生源から最終収集場所までの最短経路上に絞り，図 7 に示すような入力経路の合流数が多いノードから順に探索を行うことで，より効果の高い配備先から探索することができ，探索時間の短縮に繋がる．

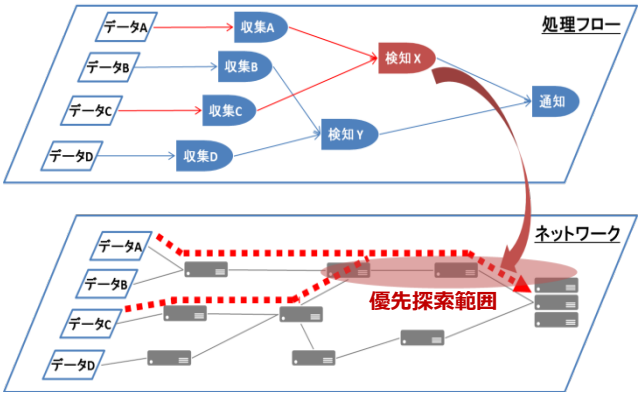



図 7 ネットワーク経路の合流点を優先した探索

② 処理のデータ入出力量比に応じた探索

入力データ量に対する出力データ量の割合が小さく，データ圧縮率が高い処理は，一部のデータを迂回させても，データ発生源に近いノードに配備してデータを圧縮してから送信することで，全体のトラフィック量を削減することができる．

そのため，入出力量比が一定以上の処理については，図 8 に示すようなデータ発生源からの距離が近いノードから順に探索を行うことで，データ圧縮率を生かした配備先を迅速に見つけることができる．

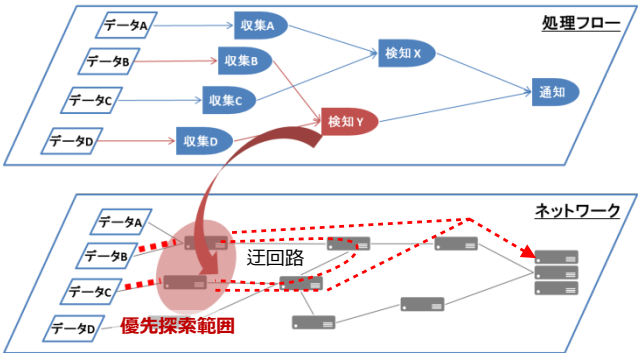



図 8 データ発生源からの距離を優先した探索

これら 2 つ探索方法を使い分ける入出力量比の閾値は，各処理の準最適解を見つけるまでの探索回数をカウントすることで常時増減を行い，アプリケーションの特徴に合わせて最適な閾値となるようにする．

### 5.2 監視トラフィックの削減

外部環境の変化に素早く追従して再配備を行うためには，監視情報を頻繁に送信する必要がある．しかし図 9 に例示するように，頻繁な監視情報の送信は，分散配備による収集トラフィック削減の効果を相殺してしまう．

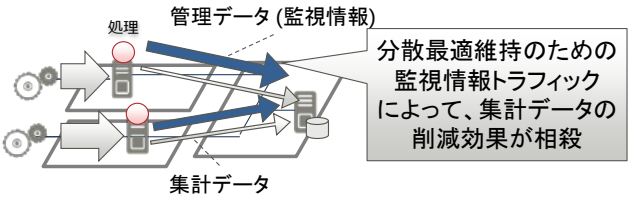



図 9 監視情報トラフィック増加の課題

分散サービス基盤は，監視データの送信に通知条件を設けて，配備最適化に必要ないデータ送信を抑制することで，素早い追従と省トラフィック性を両立させる．配備最適化処理の中で，各処理の配備先が変わる集計データ量を閾値として計算し，通知条件に用いる．この閾値は，最適な配備組み合わせの探索時に，評価値が最適になる配備組み合わせに加えて，評価値が二番目に最適となる組み合わせの評価値を記憶し，両組み合わせの評価値の大小が入れ替わる通信量を計算する．

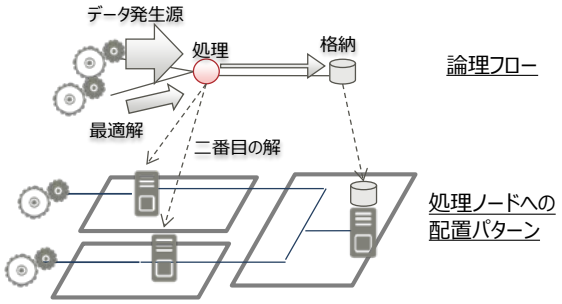



図 10 配備先の最適解と二番目の解の例

閾値の決定方法を図 10 の例を用いて説明する．この例では，二つのデータ発生源を持つ処理を上経路のノードとした経路のノードの二つの選択肢がある．図の状態は，上経路を通るデータ量が多いため，下経路を通るデータを迂回させた方が，コスト評価値が小さく望ましいケースである．つまり，対象の処理を上ノードに置いた最適解の評価値が，下ノードに配備した二番目の解の評価値より小さい．

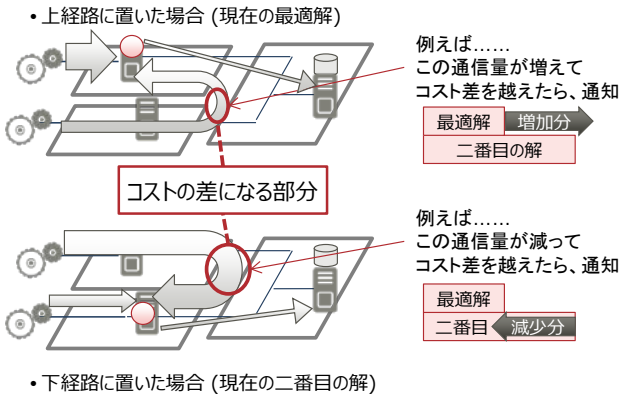



図 11 評価値 (コスト) の差と通知閾値の関係

この後，上経路を通るデータ量が増加して，図 11 上段の黒矢印で示した増加分のように上記評価値の差を越えた場合，最適解は下経路に配備するパターンに変わる．一方，下経路を通るデータ量が減少して，図 11 下段の黒矢印で示した減少分のように上記評価値の差を超える場合も，最適解は下経路に配備するパターンとなる．このように，評価値差とネットワークコストの値から，各通信量の変動によって最適解が変わる閾値を求めることができる．

実際には両経路の通信量は独立に変動するため，例えば図 11 の例では，上経路の通信量が増加すると同時に下経路の通信量が減少した場合，両閾値を越える前に最適配備先は下経路上のノードに移る．また，他の処理や通信データとノード処理上限，通信量上限などの制約によっても，最適配備先の変化は起きる．このため，上記で算出した値を目安として，監視データ量の許容範囲に合わせて通知閾値を調整する．

この手法により，再配備に影響しやすい変化を優先して監視サーバに通知することができ，既存技術より小さな監視トラフィックで通信量変化への追従性能を維持できる．

## 6. 評価

### 6.1 評価システム

前章のスケーラビリティを向上させる提案手法の効果を確認するため，疑似的なデータ収集システムを構成し，最適配備時間と監視トラフィックの評価を行った．

評価システムはデータ発生源がある拠点エリアと中継

エリア，データセンターエリアを VM 上の仮想サブネットで構成し，各エリア間の通信に疑似的な遅延と転送上限を与えて広域分散システムを模擬した．処理ノードと管理ノードは各エリアに対応するサブネット上の VM として構成し，アプリトラフィックと管理トラフィックを分けて計測した．各 VM は，下記の性能の計算機を用いて利用した．

- VM サーバハードスペック：
  - CPU：Xeon E5-2670 (2.60GHz, 8 core) x2, HT=ON（計 16 core，32 thread）
  - メモリ：64GB
- HyperVisor：
  - OS：Linux（Ubuntu 12.04，64bit）
  - IaaS 環境基盤：OpenStack（Essex）
- VM：
  - VM：KVM
  - OS：Fedora 16

管理ノードは 4VM で構成し，配備処理最適化を担当する VM には KVM の 1 仮想 CPU (2.6GHz) と 5GB のメモリを，転送経路計算と分散配備実行，監視データ収集を担当する 3VM には，それぞれ 1 仮想 CPU (2.6GHz) と 1GB のメモリを割り当てた．また，拠点・中間・集計の各処理ノードには，いずれも 1 仮想 CPU (2.6GHz) と 1GB のメモリを割り当てた．

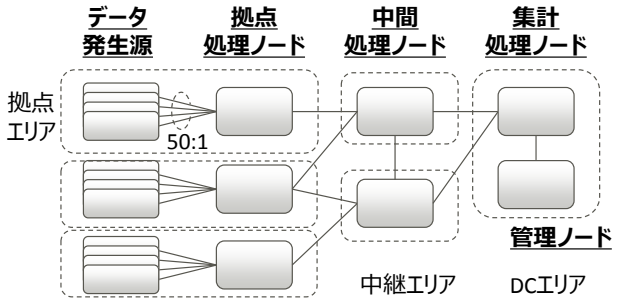



図 12 評価システムの構成

図 12 に，データ発生源から処理ノードを介してデータセンターの集計処理ノードまでのネットワーク構成を示す．

各拠点上のデータ発生源数は 50 とし，顧客拠点数は各評価に合わせて 2～300 に変更して，配備最適化時間，配備実行時間，収集と監視の両トラフィック量を計測した．ネットワーク構成はインフラ層の管理機構からの取得を主として行うが，通信コストなど自動取得が困難なデータは，実験系の構築に合わせてサブネット間の関係に応じた値をルールによって設定して用いた．

### 6.2 配備最適化時間評価

本節では，5.1 節で示した手法の効果を評価するため，与えられた処理フローを処理内容と入力データで分割し，各部分処理の最適な配備先を計算する時間を計測した．以下，アプリケーションを最初に分散配備する際の配備最適化時間を評価する実験 1 と，一定数の機器に対応した処理を分散配備済みの状態で，機器数が増加した時の最適化時間を評価する実験 2 について，説明する．

✧ 実験 1: 初期配備最適化

この実験では，処理ノード上にアプリケーションが配備されていない状態から，最適な配備先を計算するまでの時間と最適解の評価値を計測し，既存の探索方式と比較して 5.1 節の探索方法の効果を評価する．

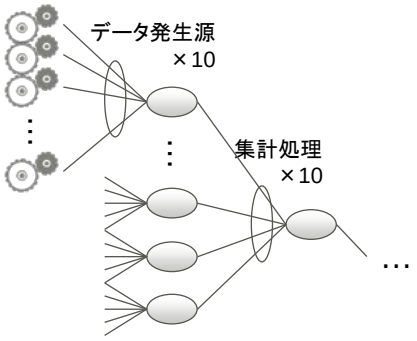



図 13 実験 1: 分割した処理間の入出力フロー構成

実験 1 の分散対象のアプリケーションは，図 13 のようなデータ発生源からの出力を多段の 10 分木で集約 (10 種の ID の入力データを集約処理し，1 つの集計 ID を持つデータを出力する) する処理フローとした．集計段数は 2～4 とし，処理数 30, 300, 3000 の 3 パターンに対して分散最適化処理を実行した．また，各処理のデータ圧縮率を表す入出力量比は，10%～90%までランダムに割り当てた．

処理ノードは処理負荷上限を一定にし，配備処理数に合わせて，処理ノードの数を 100, 1000, 10000 として規模の拡大に伴う計算時間の増加傾向を評価した．この時，ネットワーク構成情報として，各ノードが置かれたエリア間の距離に応じた従量型のコストを与えた．

- 既存方式: 各部分処理について，ランダムに配備先ノードを割り当てて比較しつつ探索する方式
- 提案方式: 5.1 節の工夫で，各処理の配備先ノードの探索開始位置と範囲を決定して探索する方式

この評価では, 前述した 3 種の処理･ノード群について，配備の最適化にかかる時間と，4.4 節で述べた評価関数の評価値を計測し，後者は既存方式と提案方式の比をグラフに示した．評価値は総通信コストに相当する値を用いており，評価値が小さいほど，同じ処理を低通信コストで実現する，効率的な分散であることを示す．

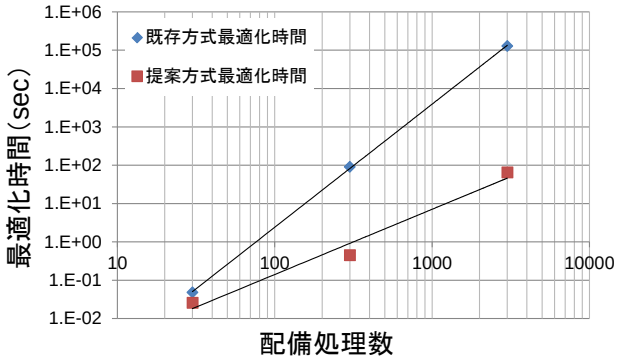



図 14 全体配備最適化時間評価

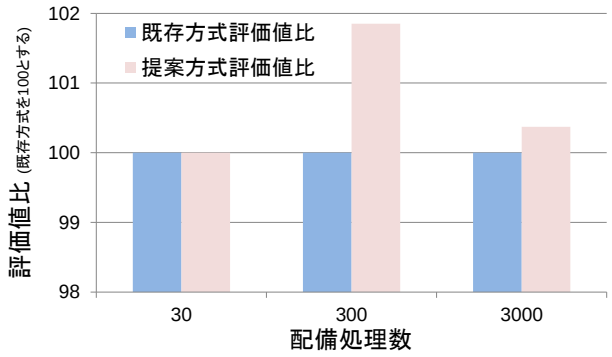



図 15 全体配備 既存方式に対する評価値比

図 14 に既存方式と提案方式の最適化時間を線グラフで，図 15 に提案方式による最適化結果評価値の対既存方式比を百分率にして棒グラフで表す．既存方式では安定解を見つけるのに 35 時間以上かかる処理数 3000 の環境において，提案方式では評価値の悪化を 1%程度に抑えた組み合わせを約 65 秒で計算することができた．総通信コストに相当する評価値は，提案方式の方が若干大きな値となっており，既存方式に対して最大で 102%ほどであった．

✧ 実験 2: 機器追加追従最適化

この実験では，一定数の機器に対応するアプリケーションが処理ノード上に配備済みの状態から，同じ割合で機器を追加し，対応する追加処理の最適な配備先を計算するまでの時間を計測して追従性能を評価する．これは，サービスをスモールスタートさせた後に機器が増加するような環境変化に相当する．

この実験では，6.1 節で示した拠点 10 ヶ所ごとにデータ発生源を 1 台ずつ増加させることによって，0.2%の機器増加に対する処理の最適化時間を測っている．この実験では，拠点数を 10～80 まで増加させて，同じ割合の機器増加に対する最適化時間の増加傾向を評価した．

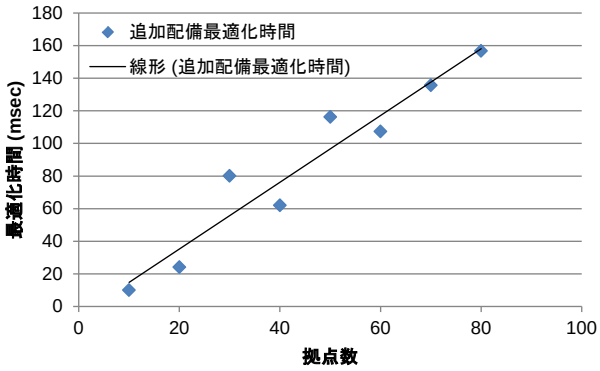



図 16 追加配備最適化時間

図 16 に，提案方式による 0.2%の機器追加に対応する追加処理の配備先の最適化計算処理に要する時間を示す．全体規模の増加に対して，最適化時間もほぼ線形に増加しており，10 拠点毎に約 20msec 増加する傾向が得られた．

### 6.3 配備実行時間評価

本節では，前節で評価した配備最適化に続いて，決定した配備先への処理の配備が終わるまでの時間を計測した．前節の実験 1 のように，処理ノードへの初めてのアプリケーションの分散実行時間に相当する．

処理の配備先となる処理ノードは，管理サーバから直接 IP 経由でアクセス可能になるように，各拠点 LAN から VPN でセンター内のセグメントに接続している．部分処理の配備や設定の更新は，OpenSSH 5.8p1 の scp によるデータ転送と ssh によるリモート制御で実行した．配備する処理のサイズは 1kB～20kB の範囲であった．

この実験では，6.1 節に示す 50 個のデータ発生源を持つ拠点の数を 4～16 まで変化させて実行時間を計測し，拠点数・処理ノード数・データ発生源数を比例関係で増加させた際の実行時間傾向を計測した．各処理ノードへの配備はシーケンシャルに行い，データの移動や制御指示は並列化していない．前節の最適化時間の規模 (～80 拠点) に比較して規模が小さいのは，実験のために準備する配備先ノード数に限度があるためである．

測定した実行時間の内訳を，以下に示す．

- 配備最適化時間
- 転送ルール作成時間
- 処理・処理設定配信時間
- 転送ルール配信時間

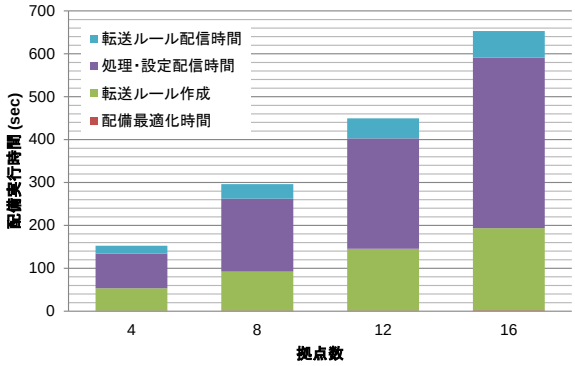



図 17 初期配備実行時間評価結果

図 17 が，各拠点数での配備実行時間の結果である．この規模までは，処理や設定の配信時間が半分以上を占めている．拠点数 16 までの計算結果から，最適化以外の配備処理時間は，いずれも拠点数に比例する時間で実行可能であることが確認できた．

### 6.4 監視トラフィックコスト評価

この節では，アプリケーション実行のためのアプリトラフィックの量と，分散配備最適化のために取得する監視トラフィックの量を計測した．

前節までのセンサーと拠点の関係に於いて，各センサーが 2 秒に 1 回出力するデータ量を 200～300bytes に正弦波状に変動させ，集計処理を自動再配備させて収集と監視のトラフィックコストを計測した．センサーごとに正弦波の周期と位相を変えて，最適な集計場所が随時変動するようにしている．監視データは 30 秒に 1 度収集し，配備最適化処理に用いる．各実験の計測時間は各正弦波が平均 2 周期入る時間とした．センサー情報の収集トラフィック量には不確定性があり，全実験を通じて±20%ほどの幅があった．下記のグラフは，各拠点数での 4 回の計測結果を平均した結果を示している．

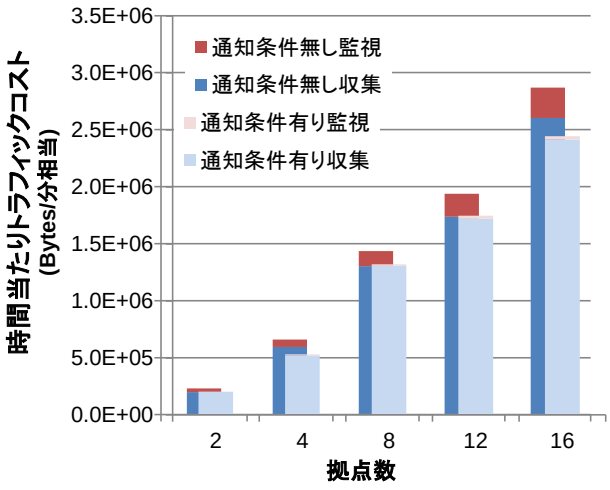



図 18 通知条件による時間当たりトラフィックコスト比較

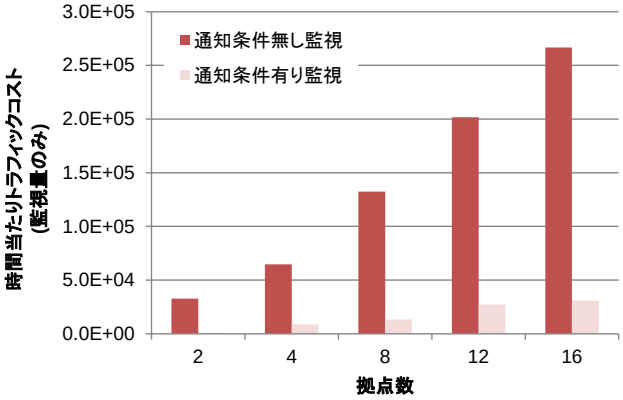



図 19 監視データのみの比較

この実験の結果を表すのが，監視・収集の総トラフィックコストを示した図 18 と，この内監視データのトラフィックコストのみを示した図 19 である．図 19 に示すグラフのように，配備最適化処理から作成した通知条件を与えることで，監視トラフィックを 1/10 程度に大きく削減したことが確認できる．図 18 の総コストでも，従来の常時監視する通知条件無しのコストに対して，通知条件有りのコストがすべての拠点数で下回っている．

### 6.5 考察

この節では，前節までの 3 種 4 実験の結果を元に提案する分散サービス基盤の性能と効果を考察する．

配備の最適化時間評価からは，提案手法はシンプルな組み合わせ探索に比べて大幅な処理時間の短縮を実現しており，処理数 3000 の段階では 1000 倍の高速化となっている．また，現実的な最適化時間の目安として，一つの処理ノードへの配備実行時間と同等の，十秒オーダーで最適化可能な規模数を比べると，既存方式の処理数 150，拠点数 3 程度に比べ，提案方式は処理数 1500，拠点数 30 程度まで最適化が実行でき，同等のサーバコストで約 10 倍の規模のシステムを処理可能である．

また，提案方式による機器数増加時の追従性能は，追加・変更ノード数にほぼ比例した時間で最適化処理が可能であった．この結果から，少しずつ変化していく収集システム環境に対しても，指数時間のような非現実的な計算時間をかけずに追従が可能であることを示した．

配備実行時間に関して，拠点数 16 程度の小規模構成時に主に時間を必要としていたのは，処理やルールの配信時間であった．規模を拡大する際には，処理ノードごとの配信を並列化することで配信時間を短縮することが重要になる．今回の計測結果から，各処理時間を線形に近似した式を用い，処理ノード 50 台毎に配信サーバを 1 台ずつ設けたと仮定して，より大規模のデータ発生源に対する機器追加への追従時間の試算を行った．

環境変化の内容は，6.2 節の実験 2 の内容に準じ，機器 25 万台 (拠点数 5000) から 100 万台 (拠点数 20000) に対して，0.2%の機器を追加配備させることとした．このケースに対する追従時間の試算結果が，図 20 である．追加処理の配備最適化時間や転送ルール作成時間は，小規模実験時の値を拠点数に対して線形に増加するとして試算値を求めた．追加処理と設定，及び転送ルールの配信時間は，並列に指示を受けた 100～400 台の配信サーバが 50 拠点に配信を行うものとして所要時間を計算している．

試算の結果, 拠点数 20000, 機器 100 万台に対して 2000 台の機器を追加するケースでも，対応する追加処理を 5 分以内に配備完了できる見込みを得た．

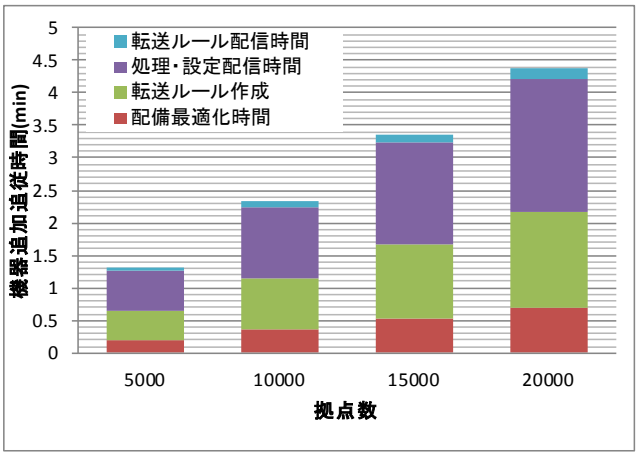



図 20 大規模構成時の機器追加追従時間試算

監視トラフィックコスト評価については，データ発生源のデータ生成量が独立に変化する事例に於いて，出力データ量変動への最適配備状態の追従を維持しながら，監視データ量を大きく削減可能であることを確認した．監視データの通知を抑制する影響でアプリケーションの収集データ量は若干増加すると予想されたが，この実験の例では収集データ量の変動幅の範囲内に収まっており，すべての実験結果で総トラフィックの削減を確認できた．

以上の結果により，分散最適化処理を高速化することで，100 万台機器が 0.2%増加しても数分で再配備を可能な見込みを得た．また，常時変動する収集環境の監視データ通信量を小さくすることで，分散運用ライフサイクル全体の通信量を削減可能なことを示した．

## 7. おわりに

本論文では，実世界の環境変化に追従して自動分散配備を行う分散サービス基盤を提案した．また小規模実験と机上計算から，分散配備最適化と分散実行監視を効率化し，大規模化したシステムにおいても高速な分散配備と追従が可能となることを示した．

今回の実験では，特に処理の入出力量に着目した分散配備最適化を行った．今後は，データ量に比例しない処理負荷のフィードバックやレスポンスへの影響など，より正確な環境変化の検出と，それに応じた分散最適化を実施していく．また，実際のシステムに適用して評価を行い，システム全体の管理コスト削減効果を確認していく．さらに，収集データ量や処理負荷の変動履歴から環境変化を予測し，より迅速な分散状態の更新を可能にすることを目指す．

## 参考文献

1) a Shneidman, Jeffrey, et al. "A cost-space approach to distributed query optimization in stream based overlays." Data Engineering Workshops, 2005. 21st International Conference on. IEEE, 2005.
2) a Abadi, Daniel J., et al. "The Design of the Borealis Stream Processing Engine." CIDR. Vol. 5. 2005.
3) a Borealis Distributed Stream Processing Engine, http://cs.brown.edu/research/borealis/public/
4) a Hwang, Jeong-Hyon, Ugur Cetintemel, and Stan Zdonik. "Fast and reliable stream processing over wide area networks." Data Engineering Workshop, 2007 IEEE 23rd International Conference on. IEEE, 2007.
5) 福田茂紀，福井誠之，中川格，佐々木和雄 “センサ情報収集ノードへのアプリケーション処理分散” 電子情報通信学会通信ソサイエティ大会, BS-4-7, 2011.
6) Nomura, Yoshihide, et al. "Massive event data analysis and processing service development environment using DFD." Services (SERVICES), 2012 IEEE Eighth World Congress on. IEEE, 2012.
7) Kimura, Kosaku, et al. "Multi-query Unification for Generating Efficient Big Data Processing Components from a DFD." Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on. IEEE, 2013.